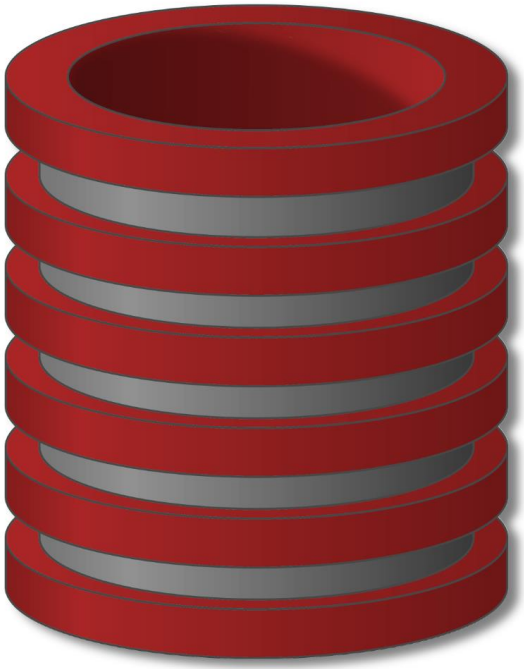


Transakcije

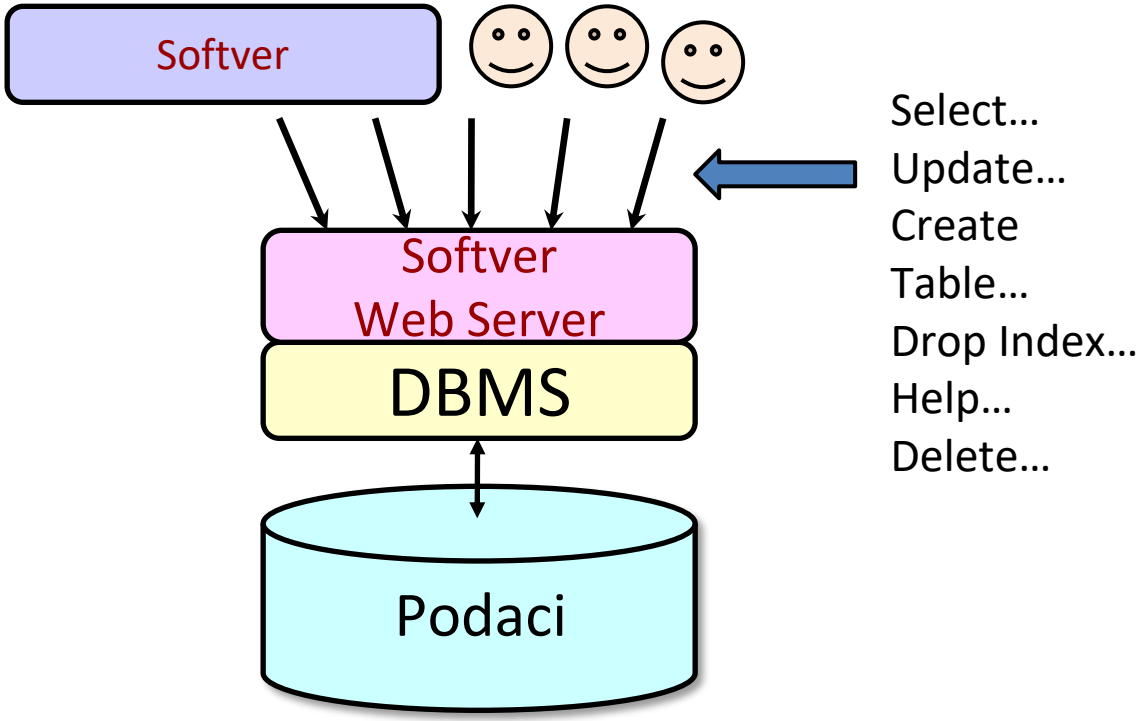
Predmet: Administriranje Baze Podataka

Predavač: dr Dušan Stefanović

Potreba za transakcijom

- Istovremeni pristup bazi podataka
- Otpornost na greške u sistemu

Istovremeni pristup bazi podataka



Istovremeni pristup:
Nekonzistentnost na nivou atributa

Update Fakultet
Set BrojMesta = BrojMesta+ 1000
Where Fime = 'VTS'

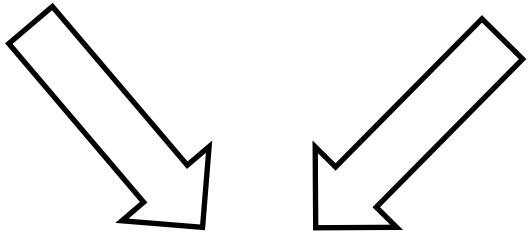
Update Fakultet
Set BrojMesta = BrojMesta+ 1500
Where Fime = 'VTS'

Fime	grad	BrojMesta
VTS	Nis	300
ETF	Beograd	450
Teh. Fak.	Novi Sad	430
ELFAK	Nis	500

Fakultet

Istovremeni pristup:
Nekonzistetnost na nivou vrste

Update Prijava
Set Smer = ‘SRT’
Where sID = 5



Prijava

Sid	Fime	Smer	Odluka
1	VTS	SRT	*Primljen*
~~1~~	~~ETF~~	~~RTI~~	~~Odbijen~~
~~5~~	~~ELFA K~~	~~KOT~~	~~Odbijen~~
2	FON	SRT	Odbijen

Update Prijava
Set Odluka= ‘Primljen’
Where sID = 5

Istovremeni pristup:
Nekonzistetnost na nivou tabela

```
Update Prijava
Set Odluka = 'Primljen'
Where sID In(Select sID
             From Student
             Where Prosek > 3.9)
```

```
Update Student
Set Prosek = (1.1) * Prosek
Where Vskole  > 500
```

Prijava

Sid	Fime	Smer	Odluka
4	VTS	SRT	*Primljen*
1	ETF	RTI	Odbijen
3	ELFAK	KOT	Odbijen
2	FON	SRT	Odbijen

Student

Sid	Sime	Prosek	Vskole
1	Marko	3.8	*580*
2	Darko	4.3	400
3	Jelena	4.9	620
4	Darko	3.6	300

Istovremeni pristup:
Nekonzistetnost na osnovu višestrukih iskaza

```
Insert Into Arhiva
  Select *
  From Prijava
  Where Odluka = 'Odbijen';

Delete From Prijava
Where Odluka= 'Odbijen';
```

```
Select Count(*) From Prijava;
Select Count(*) From Arhiva;
```

Prijava

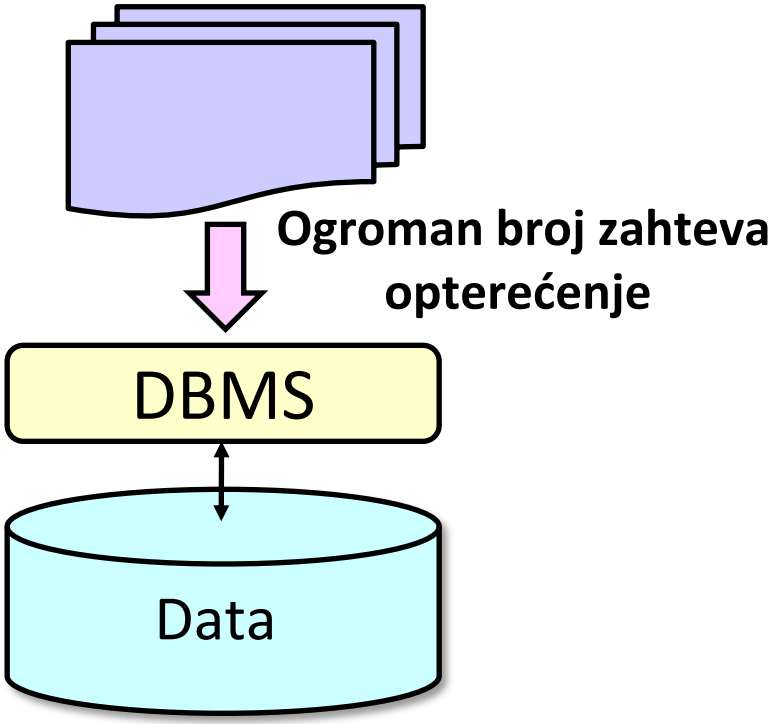
Sid	Fime	Smer	Odluka
4	VTS	SRT	*Primljen*
1	ETF	RTI	Odbijen
3	ELFAK	KOT	Odbijen
2	FON	SRT	Odbijen

Izvršavanja više iskaza istovremeno

- Jednostavno rešenje: Izvršavanje SQL iskaza u izolaciji

  Želimo da obezbedimo višestruki pristup uvek kada je to bezbedno

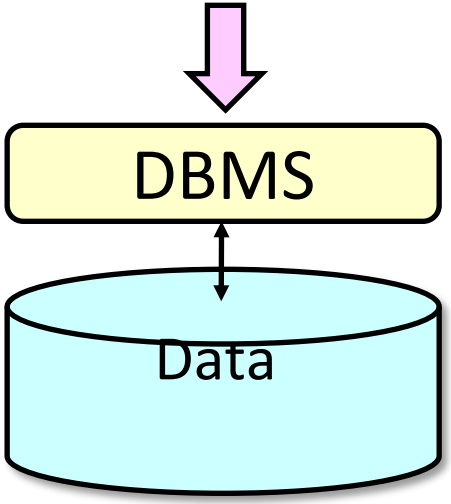
Otpornost na sistemske otkaze



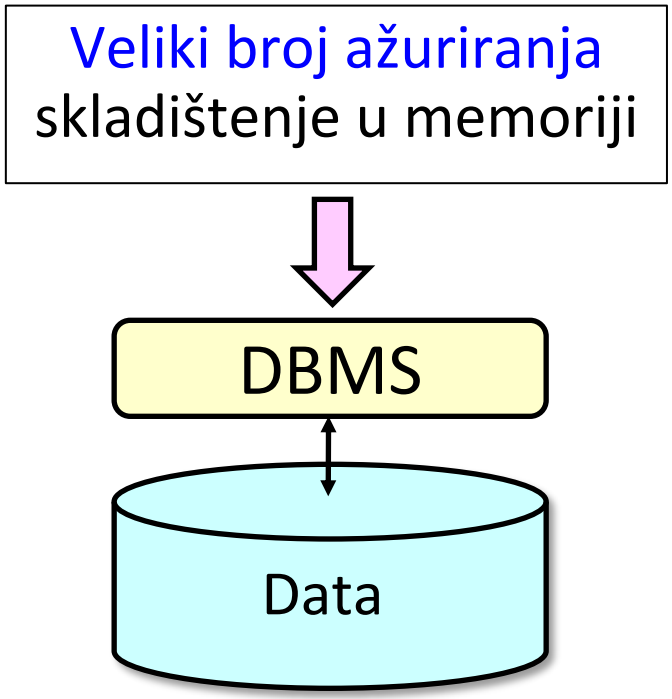
Otpornost na sistemske otkaze

```
Insert Into Arhiva
  Select *
  From Prijava
  Where Odluka = 'Odbijen';

Delete From Prijava
Where Odluka= 'Odbijen';
```

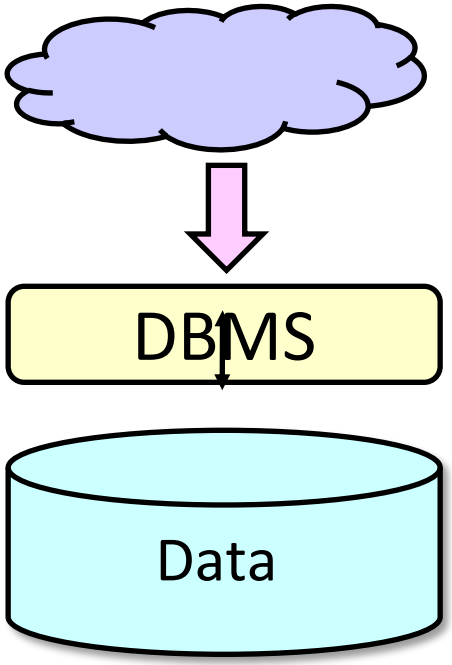


Otpornost na sistemske otkaze



Sistemske greške --: Cilj

Garancija izvršenja operacije u potpunosti, u suprotnom operacija se ne izvršava



Rešenje za istovremeni pristup i sistemske otkaze

Transakcije

Transakcija je skup jedne ili više SQL operacije koje se tretiraju kao celina

- Transakcija se izvršava u izolaciji
- Ako sistem otkaže, svaka transakcija će se izvršiti u potpunosti ili se uopšte neće izvršiti

Rešenje za istovremeni pristup i sistemske otkaze

Transakcije

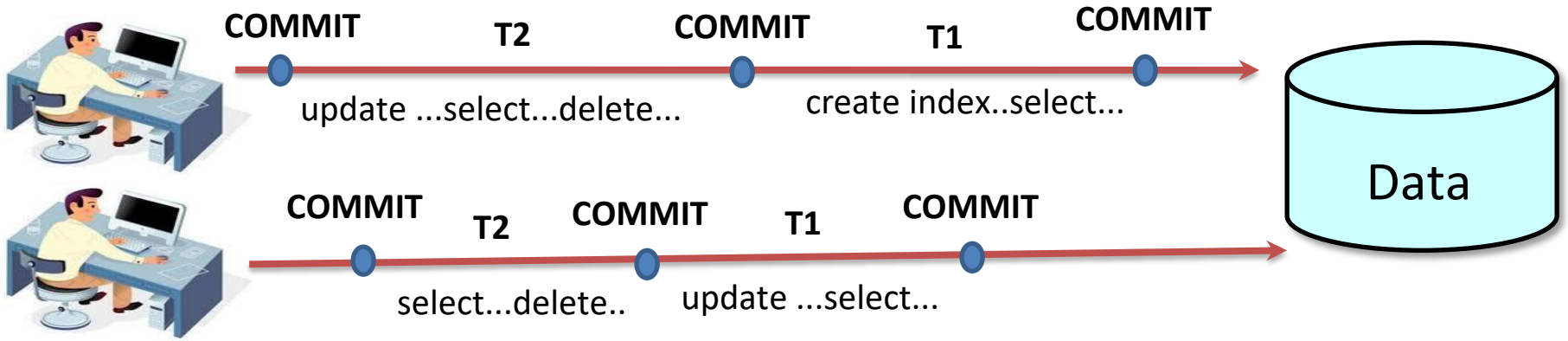
Transakcija je skup jedne ili više SQL operacije koje se tretiraju kao celina. SQL standard:

- Transakcija počinje prvim SQL iskazom
- Specijalnom komandom "**commit**" transakcija se završava i počinje nova
- Trenutna transakcija se završava i kada se prekine sesija sa bazom
- "**Autocommit**" mod svaki SQL iskaz se izvršava kao transakcija

Rešenje za istovremeni pristup i sistemske otkaze

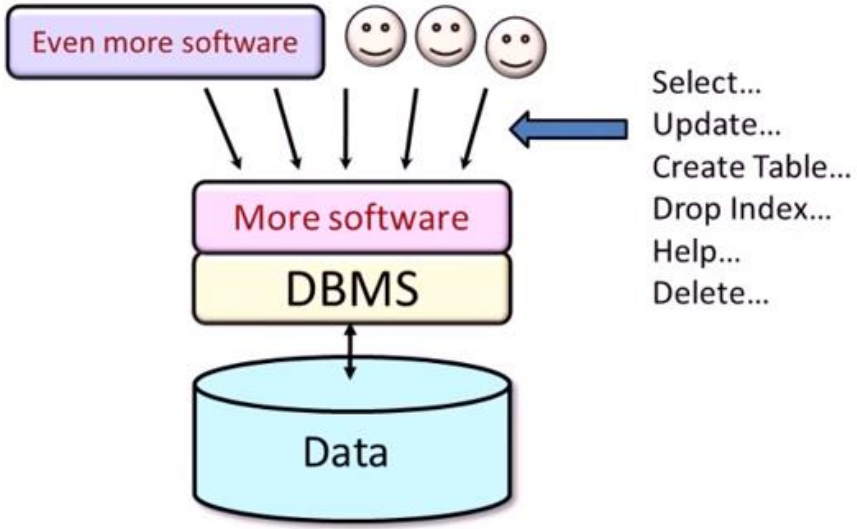
Transakcije

Transakcija je skup jedne ili više SQL operacije koje se tretiraju kao celina. SQL standard:



MOTIVACIJA ZA POJAVU TRANSAKCIJA

- **Konkuretnost** – istovremeni pristup bazi od strane više korisnika
- **Otpornost na otkaz sistema**



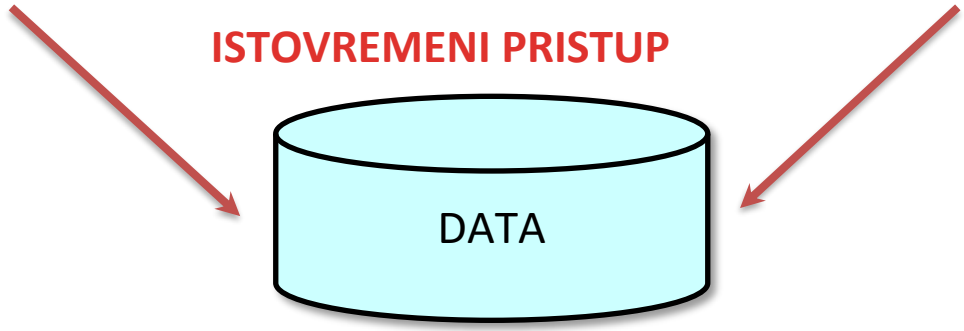
Uzastopni pristup: Nekonzistentnost na nivou atributa

T1

```
Update Fakultet
Set BrojMesta = BrojMesta+ 1000
Where Fime = 'VTS'
```

T2

```
Update Fakultet
Set BrojMesta = BrojMesta+ 1500
Where Fime = 'VTS'
```



Varijanta1: T1, T2
Varijanta2: T2, T1

Uzastopni pristup: Nekonzistetnost na nivou reda (konfliktna situacija)

T1

```
Update Prijava
Set Smer = 'SRT'
Where sID = 5
```

T2

```
Update Prijava
Set Odluka= 'Primljen'
Where sID = 5
```

Sid	Fime	Smer	Odluka
1	VTS	SRT	*Primljen*
~~1~~	~~ETF~~	~~RTI~~	~~Odbijen~~
~~5~~	~~ELFAK~~	~~KOT~~	~~Odbijen~~
2	FON	SRT	Odbijen

Varijanta1: T1, T2

Varijanta2: T2, T1

Istovremeni pristup:
Nekonzistetnost na nivou tabela

T1

```
Update Prijava
Set Odluka = 'Primljen'
Where sID In(Select sID
             From Student
             Where Prosek > 3.9)
```

T2

```
Update Student
Set Prosek = (1.1) * Prosek
Where Vskole  > 500
```

Prijava

Sid	Fime	Smer	Odluka
4	VTS	SRT	*Primljen*
1	ETF	RTI	Odbijen
3	ELFAK	KOT	Odbijen
2	FON	SRT	Odbijen

Student

Sid	Sime	Prosek	Vskole
1	Marko	3.8	*580*
2	Darko	4.3	400
3	Jelena	4.9	620
4	Darko	3.6	300

Varijanta1: T1, T2

Varijanta2: T2, T1

Redosled je vrlo bitan posebno za studenta sa SID=1

Istovremeni pristup:
Nekonzistetnost na osnovu višestrukih iskaza

T1

```
Insert Into Arhiva
  Select *
  From Prijava
  Where Odluka = 'Odbijen';

Delete From Prijava
Where Odluka= 'Odbijen';
```

T2

```
Select Count(*) From Prijava;
Select Count(*) From Arhiva;
```

Prijava

Sid	Fime	Smer	Odluka
4	VTS	SRT	*Primljen*
1	ETF	RTI	Odbijen
3	ELFAK	KOT	Odbijen
2	FON	SRT	Odbijen

Varijanta1: T1, T2
Varijanta2: T2, T1

Redosled je bitan

Nekonzistetnost na osnovu višestrukih iskaza

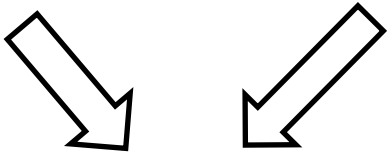
slučaj kada klijenti interaguju sa više tabela

T1

```
Insert Into Arhiva
  Select *
  From Prijava
  Where Odluka = 'Odbijen';

Delete From Prijava
Where Odluka= 'Odbijen';
```

- Prvi klijent određene podatke iz tabele Prijava premešta u tabelu Arhiva
- Svi studenti koji su odbijeni iz tabele prijava se premeštaju u tabelu Arhiva a zatim se brišu iz tabele Prijava.



Prijava

Sid	Fime	Smer	Odluka
4	VTS	SRT	*Primljen*
1	ETF	RTI	Odbijen
3	ELFAK	KOT	Odbijen
2	FON	SRT	Odbijen

Varijanta1: T1, T2

Varijanta2: T2, T1

T2

```
Select Count(*) From Prijava;
Select Count(*) From Arhiva;
```

- Drugi klijent želi da vidi broj vrsta u tabeli Prijava a zatim i u tabeli Arhiva.
- Drugom klijentu se može prikazati duplikat vrsta ukoliko je on uneo oba iskaza nakon prebacivanja podataka iz tabele prijava u tabelu arhiva.
- Idealno bi bilo da drugi klijent oba iskaza izvrši pre ili posle prvog klijenta
- U ovom primeru garantuje se samo serijsko izvršenje operacija a ne i redosled

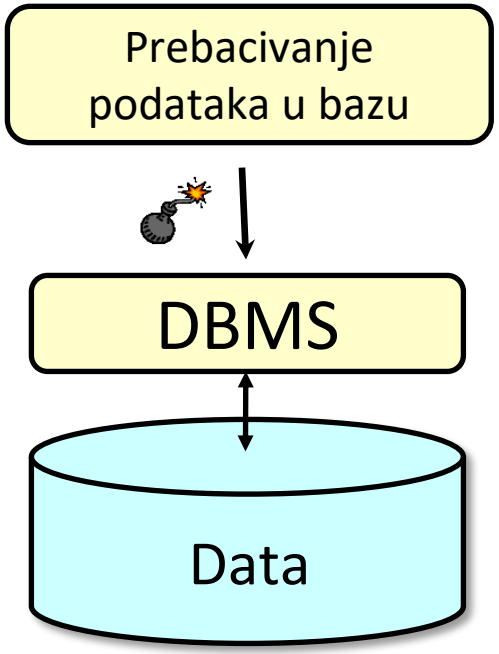
Redosled je bitan

ISTOVREMENI PRISTUP PODACIMA

- **Konkuretnost – treba obezbediti** istovremeni pristup bazi od strane više korisnika (paralelno izvršenje) ali tamo gde je to bezbedno da se uradi tj. gde se ne remeti konzistentnost podataka u bazi
  - korisnici istovremeno pristupaju različitim tabelama u bazi
- Ukoliko može da se javi konflikt potrebno je obezbediti izolaciju prilikom izvršenja SQL iskaza (serijsko izvršenje)

MOTIVACIJA ZA POJAVU TRANSAKCIJA

- **Otpornost na otkaz sistema** (System faliures)




```
Insert Into Arhiva
  Select *
  From Prijava
  Where Odluka = 'Odbijen';
```

- Otkaz sistema usled prebacivanja podataka u bazu (softverski,hardverski otkaz ili nestanak električne energije
- Pitanje šta će biti u bazi kada se sistem oporavi, šta je prebačeno a šta nije?

- Rešenje je da nam sistem garantuje sve ili ništa prilikom izvršenja upita

Rešenje za konkuretnost i nekonzistetnost prilikom otkaza sistema su

Transakcije

Atomicity

Consistency

Isolation

Durability

Rešenje za konkuretan pristup i otkaz sistema

Transakcije

Transakcija je skup jedne ili više SQL operacija koja odgovara jednoj logičkoj jedinici posla

Transakcija se izvršava u izolaciji:

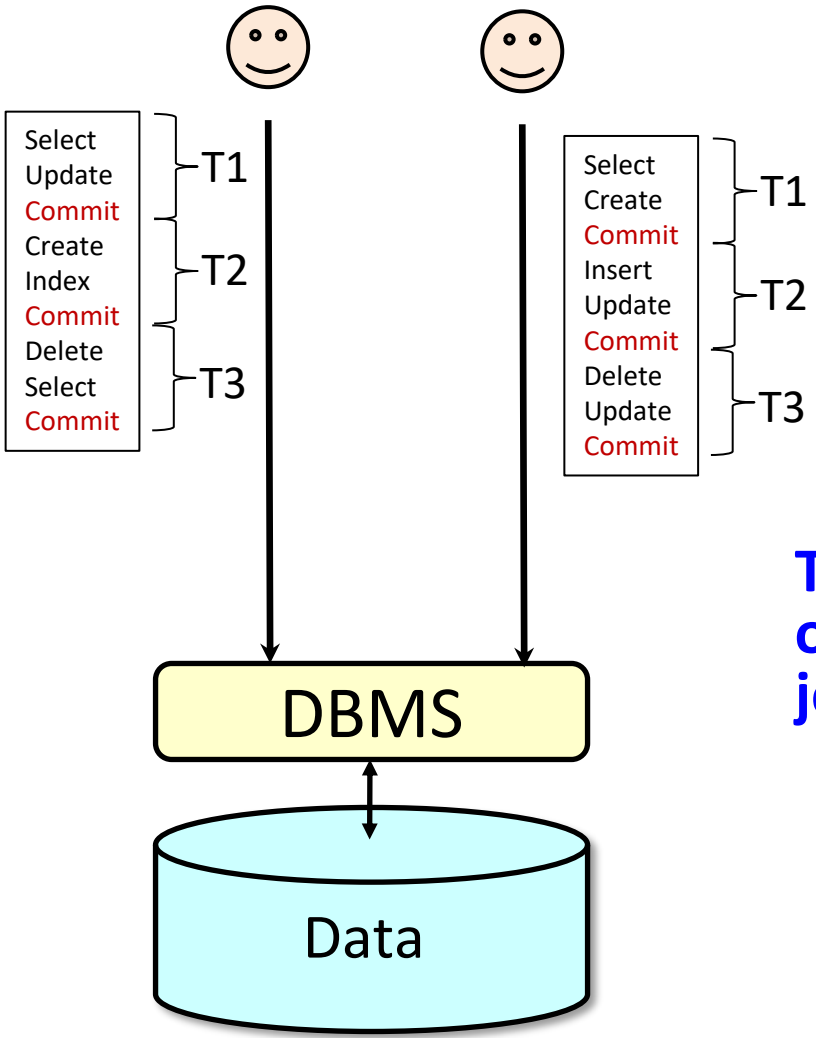
- Ukoliko više klijenata pristupaju bazi istovremeno
- Ukoliko sistem otkaže, transakcija biće izvršena u celosti ili se uopšte neće izvršiti

Rešenje za konkuretan pristup i otkaz sistema

Transakcije

Transakcija je skup jedne ili više SQL operacija koja odgovara jednoj logičkoj jedinici posla

- Transakcija počinje automatski prvim SQL iskazom
- Završetak prve i početak naredne transakcije počinje commit naredbom
- Trenutna transakcija se završava prekidanjem sesije sa bazom podataka.
- Autocommit je mod u kome svaki SQL iskaz se izvršava kao transa



Transakcija je skup jedne ili više SQL operacija koja odgovara jednoj logičkoj jedinici posla

Osobine TRANSAKCIJE

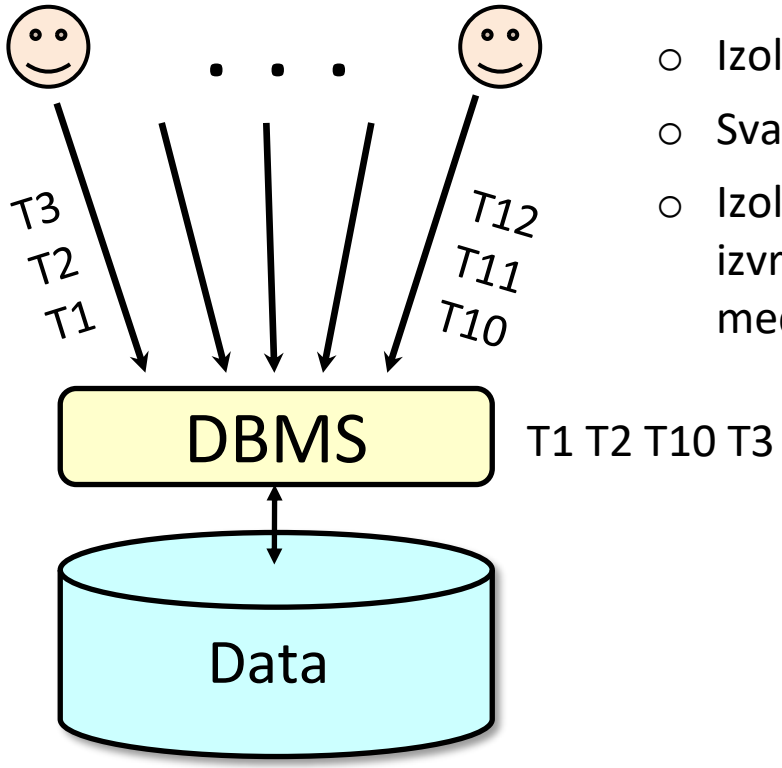
Atomicity

Consistency

Isolation

Durability

(ACID Osobine) **Izolacija**



- Veci broj korisnika istovremeno pristupa istoj bazi.
- Izolacija se zasniva na serijskom pristupu.
- Svaka transakcija predstavlja skup SQL iskaza.
- Izolacija znači da kada se dve ili više transakcija izvršavaju istovremeno, njihovi efekti moraju biti međusobno izolovani.

Serijski pristup

Operacije mogu da stižu paralelno, dok izvršenje mora da bude serijsko po nekom redosledu za sve transakcije.
Ne garantuje redosled izvršenja transakcija

ZADATAK

Klijent K1 izvršava dve transakcije T1 i T2, istovremeno klijent K2 izvršava svoje dve transakcije T3 i T4. Koliko različitih jednakih redosleda obrade ovih transakcija postoji?

a) 24

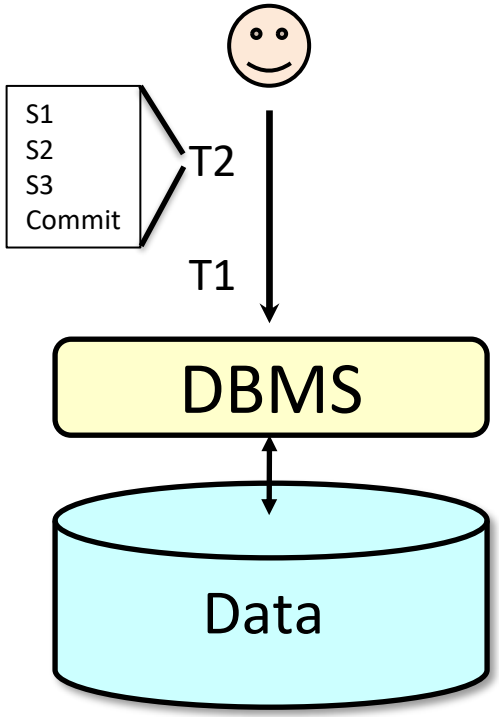
b) 6

c) 4

d) 2

Rešenje:
T1T2T3T4, T1T3T4TT2, T1T3T2T4, T3T4T1T2, T3T1T2T4,T3T1T4T2

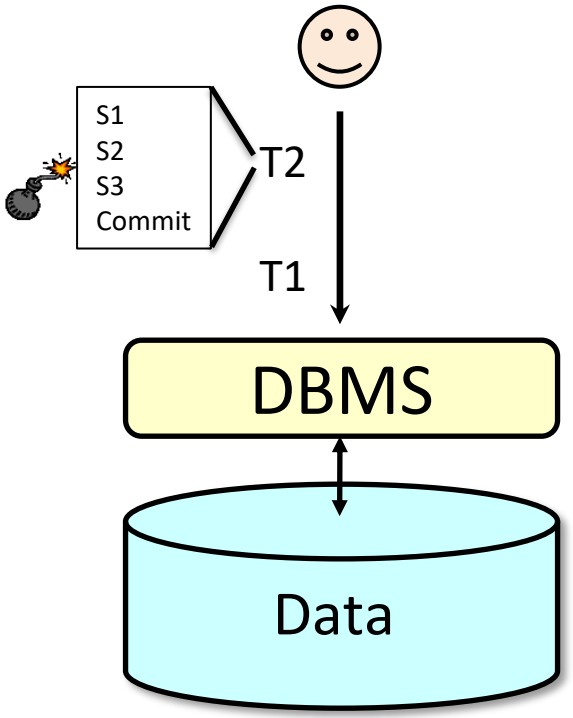
(ACID Osobine) **Izdržívost (Durability)**



Izdržívost garantuje

Ukoliko se sistem sruši nakon
potvrde transakcije, komandom commit
sve operacije transakcije biće sačuvane u bazi

(ACID Properties) **Atomnost(Atomicity)**



- U ovom slučaju otkaz u sistemu se desio u toku izvršenja transakcije a ne kao u prethodnom slučaju nakon završetka transakcije.
- Nakon oporavka, sistem će primetiti nepotpunu transakciju i obrisaće je.
- Aplikacija mora da ima mehanizam da ponovo pošalje transakciju.

Svaka transakcija je
“sve ili ništa”
ne ostavlja se nedovršenom

ZADATAK

Relacija R(A) sadrži podatke (5,6) i prosleđuju joj se dve transakcije T1 (update R set A=A+1) i T2 (update R set A=2A).Koje od sledećih iskaza nije moguć ako su ispunjeni uslovi atomnosti i izolacije.

a) (10,12)

b) (11,13)

c) (11,12)

d) (12,14)

Rešenje:
T1,T2 ... 12,14
T2,T1.... 11,13
T2 10,12
T1 ... 6,7

Svaka transakcija je
“sve ili ništa”
Nikad ne ostavlja posao na pola

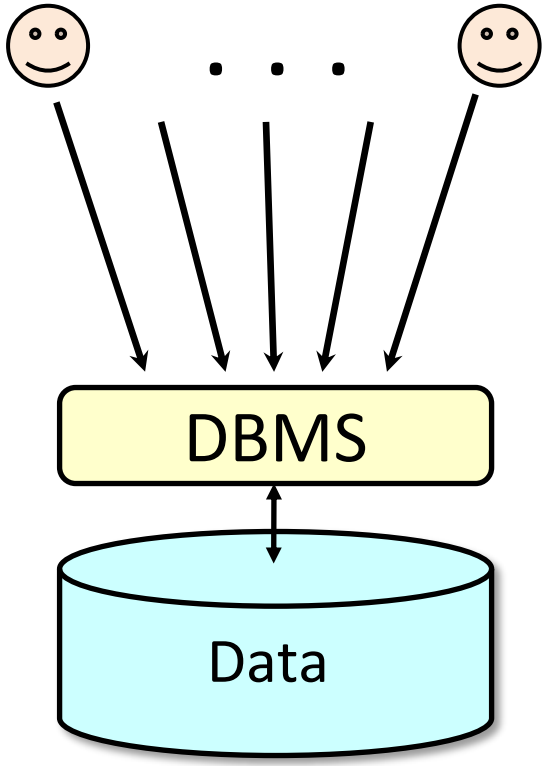
Transakcije Rollback (= Prekid)

- Poništavanje transakcije tj. vraćanje na poslednje ispravno stanje
- Može da se inicira od klijenta ili sistema kada detektuje otkaz

```
Begin Transaction;
<prosleđivanje ulaza od korisnika>
SQL komande na osnovu ulaza
<Potvrda rezlutata>
If ans='ok' Then Commit; Else Rollback;
```

Klijent inicira prekid transakcije

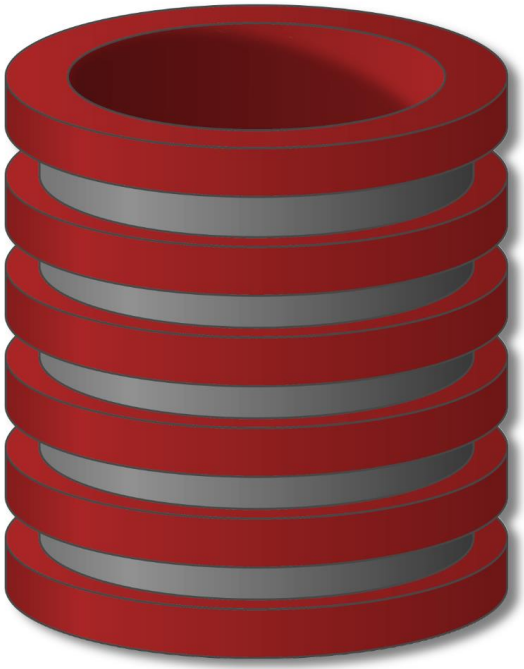
(ACID Properties) **Konzistentnost**



Svaki klijent, svaka transakcija:

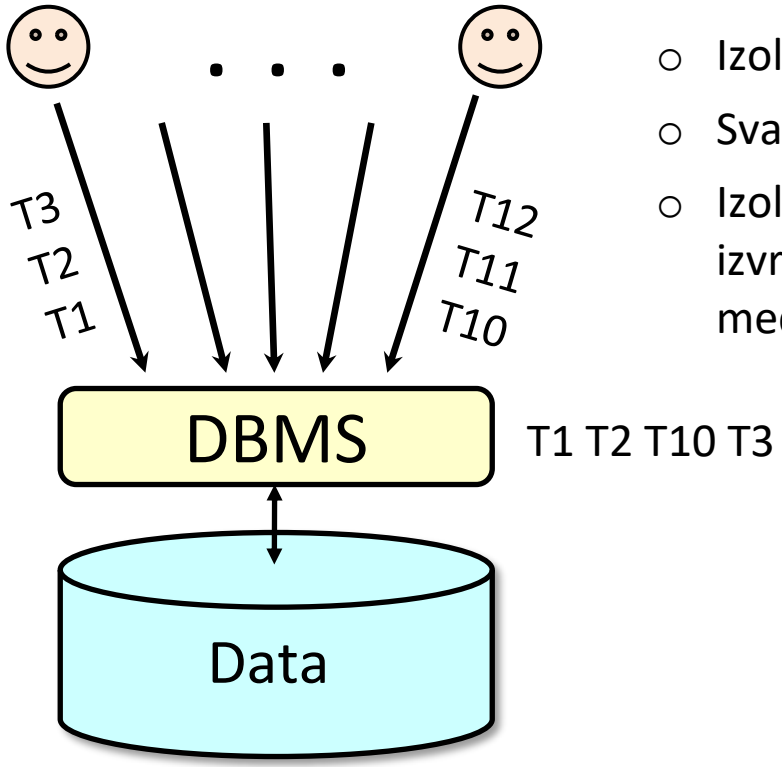
- Sva ograničenja se zadržavaju kada se izvršava transakcija

Serializability ⇒ ograničenja se uvek zadržavaju



Nivoi izolacije

(ACID Osobine) Izolacija – Protokoli zaključavanja



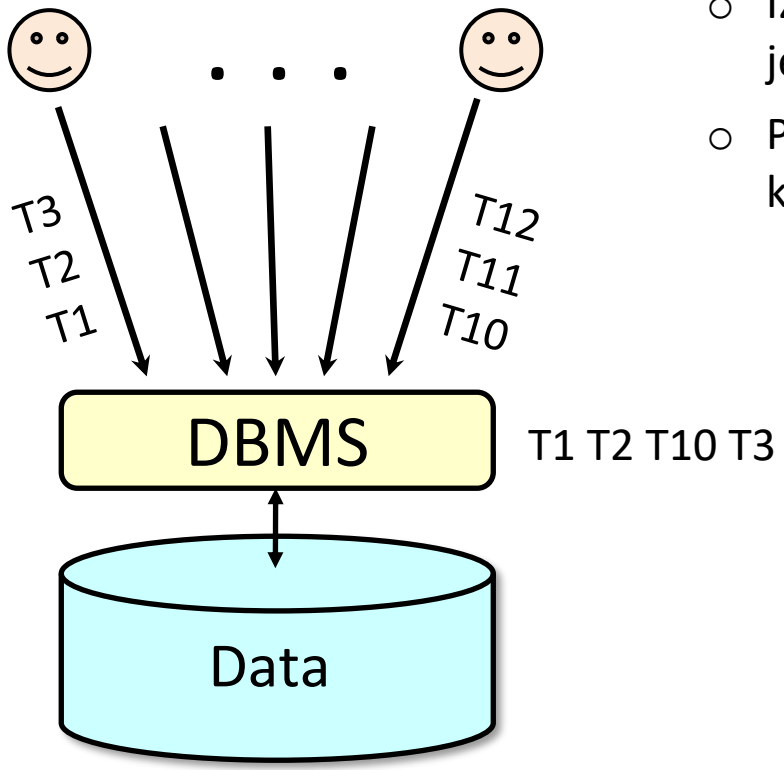
- Veci broj korisnika istovremeno pristupa istoj bazi.
- Izolacija se zasniva na serijskom pristupu.
- Svaka transakcija predstavlja skup SQL iskaza.
- Izolacija znači da kada se dve ili više transakcija izvršavaju istovremeno, njihovi efekti moraju biti međusobno izolovani.

Serijski pristup

Operacije mogu da stižu paralelno, dok izvršenje mora da bude serijsko po nekom redosledu za sve transakcije.
Ne garantuje redosled izvršenja transakcija

- Protokoli zaključavanja utiču na smanjenje konkuretnosti i na dodatni overhead.

(ACID Osobine) Izolacija – Protokoli zaključavanja



- Izolacija definiše kako i kada promene koje napravi jedna transakcija postaju vidljive drugoj transakciji.
- Protokoli zaključavanja utiču na smanjne konkuretnosti i na dodatni overhead.

Izolacijono nivoi

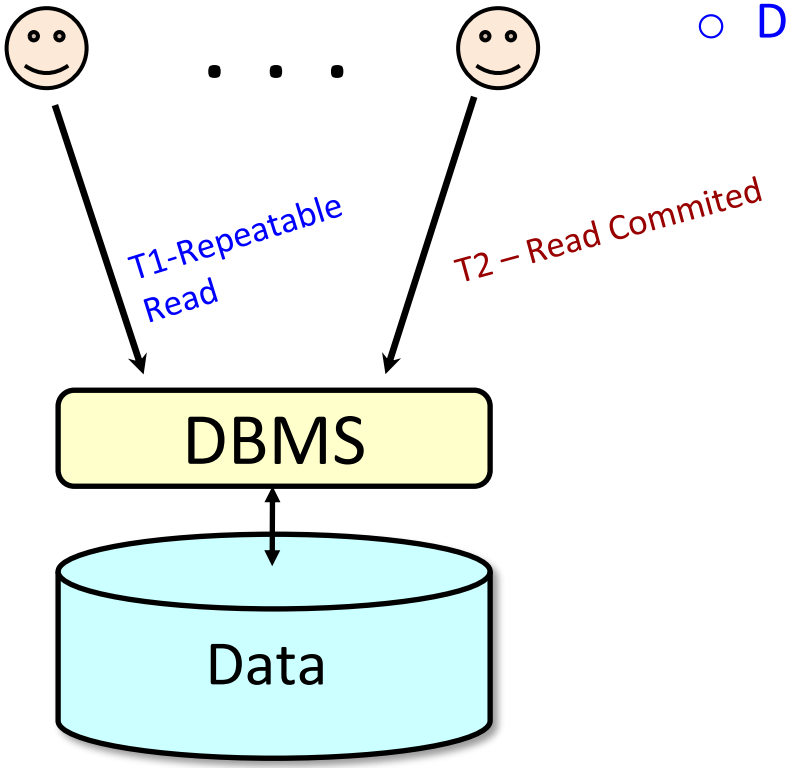
Read Uncommited

Read Commited

Repeatable Read

- Izolacioni nivoi se javljaju kao dodatni vid poboljšanja jer povećavaju konkuretnosti i smanjuju zaglavlje

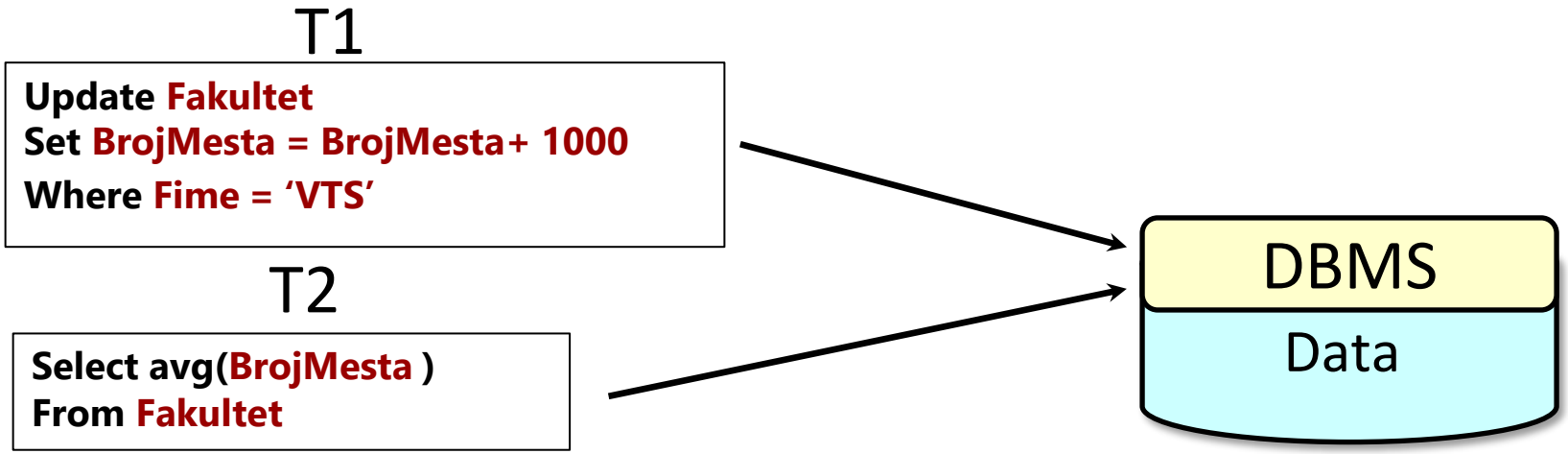
(ACID Osobine) **Izolacijoni nivoi**



- Definišu se po transakciji
  - Klijent može da podesi različit nivo izolacije za svaku svoju transakciju
  - Za dve paralelne transakcije nivo izolacije može da bude drugačiji
  - Slabiji izolacioni nivo dozvoljava da više korisnika pristupi istom podatku u isto vreme povećavajući konkuretnost (dirty reads ili lost updates).
  - Jači izolacioni nivo smanjuje konkuretnost i problemi koji prate konkuretnost ali zahteva više sistemskih resursa i povećava šansu da jedna operacija blokira drugu.

Koncept *dirty reads*

Podatak u bazi je *prljav (eng. dirty) ukoliko transakcija još nije potvrđena (commited) a druga transakcija želi da pročita(**select**) taj podatak*



Sve dok transakcija koja modifikuje podatak nije potvrđena a postoje druge transakcije koje žele da pročitaju taj podatak može da se javi problem u verodostojnosti tog podatka.

Koncept *dirty reads*

Transakcija T2 čita podatak dok transakcije T1 i T3 ažuriraju tj. menjaju podatke.

Može da se javi prljavo čitanje za kolonu prosek tj. ukoliko se T1 ne izvrši podatak koji je pročitan je irelevantan

T1 **Update Student Set Prosek = (1.1)*Prosek Where VelicinaSkole > 2500**

Konkurentan ...

T2 **Select Prosek From Student Where VelicinaSkole > 2500**

Konkurentan ...

T3 **Update Student Set VelicinaSkole=2600 Where Sid = 234**

Izolacioni nivo – read uncommited

Prvi i najslabiji izolacioni nivo

Ukoliko transakcija nema definisan izolacioni nivo default je *serializable*

Transakcija koja ima podešen ovaj izolacioni nivo može da odradi *dirty reads tj. da pročita vrednost i ako nije potvrđena*

T1

```
Update Student Set Prosek = (1.1)*Prosek Where VelicinaSkole > 2500
```

T2

```
Set Transaction isolation level Read uncommited;
Select Prosek From Student Where VelicinaSkole > 2500
```

- Transakcija T2 čita podatak koji obrađuje transakcija T1 i ako transakcija T1 nije potvrđena
- *Ne vodimo računa da li čitamo staru ili novu vrednost ali povećavamo konkuretnost i smanjujemo overhead.*

Izolacioni nivo – read commited

Transakcija ne mora da odradi *dirty reads*

Ovaj nivo transakcije je jači ali i dalje ne garantuje globalnu seriabilnost.

T1

```
Update Student Set Prosek = (1.1)*Prosek Where VelicinaSkole > 2500
```

T2

```
Set Transaction isolation level Read commited;
Select Avg(Prosek) From Student;
Select Max(Prosek) From Student;
```

Izolacioni nivo – repeatable commited

Transakcija ne mora da odradi *dirty reads*

Podatak se čita više puta

T1

```
Update Student Set Prosek = (1.1)*Prosek Where VelicinaSkole > 2500
```

T2

```
Set Transaction isolation level Read commited;
Select Avg(Prosek) From Student;
Select Max(Prosek) From Student;
```

Kontrola konkuretnosti

- Upravlja izolacijom i garantuje ispravnost
- Koristi se kod baza podataka i sistema za skladištenje podataka da bi se garantovala ispravnost izvršenja konkuretnih transakcija.
- Mehanizmi kontrole konkurencije su dizajnirani da obezbede najbolje moguće performanse u okviru ograničenja konkuretnog pristupa bazi podataka
- Dvofazno zaključavanje se koristi da se obezbedi serijski pristup.
- Da bi pristupio objektu baze podataka, transakcija mora prvo da zatraži zaključavanje objekta.
- U zavisnosti od tipa pristupa (npr., čitanja ili pisanja) i tipa zaključavanja, zaključavanja može biti blokirano i odloženo, ukoliko druga transakcija je zaključala isti objekat.